**Computer Science 220/220L**

**Laboratory 11 / Program 8**

**Hangman**

**Learning objectives:**

Develop a Python program that:

* Reads from a file
* Uses functions
* Uses decision and repetition structures to solve a problem

**Part I: Assignment:**

Write a program to support the children's spelling game hangman. Call the program **hangman.py**. Since there are many versions of hangman, be sure that you implement exactly the following.

The player is to guess the letters in a secret word. Use underscores to display the number of letters in the word. When the player guesses a correct letter, display the word with that letter showing. Correct guesses don't count against the player. Incorrect guesses count, and the player loses on the seventh incorrect guess. Each time the player is asked to enter a letter, the program should display how many guesses they have left and a list of all of the incorrect letters they have guessed. If the player accidentally chooses a letter that has already been guessed, this should not count as a guess. See sample games below.

Your program must contain at least the following functions. (If you want you may use other functions as well.)

* A function that returns a list of words read from a file. Call the file **wordlist.txt.** You must provide your own file when you test your program.
* A function that randomly picks and returns a secret word from the list.
* A function that returns the “guessed” word at a given point. (E.g. If the word is “slant” and the user has guessed the letters ‘t’ and ‘a’, this function should return “\_ \_ a \_ t”)
* A function that determines whether the letters that have been entered spell the secret word (this returns a boolean).
* A function that plays the game (i.e., other than main()).

**Part II: User Interface**

Once you get your functions working with text, you can add a graphical user interface of your design. The GUI should have the following elements:

* A text area where the word with underscores representing the letters of the word is presented.
* A text area where the letters used are displayed. Optionally you can display the letters not chosen yet. In either case you must identify which role the letters displayed serve.
* A text area indicating the number of guesses the user has remaining.
* An area where the user enters the letter s/he wishes to play.
* When a game completes, a message should appear in the GUI window asking the player whether they want to play another game. The GUI should also provide **Yes** and **No** buttons for the player to make a choice.

**Once your basic program is working:**

The traditional way to play hangman is to draw a figure on a gallows. Another part of the hanged man is added for each incorrect guess. When the figure is complete, the player loses. How you draw the figure is up to you, but it should have exactly seven parts.

Anyone who renders an image that looks like me being hanged will lose 3 points for insubordination.[[1]](#footnote-1)

**Submission:**

This lab will count as a grade for CSCI220. You should complete the assignment and submit your finished product on or before the date specified in OAKS.

1. No, not really!! [↑](#footnote-ref-1)